042390.P13483

APPLICATION FOR UNITED STATES LETTERS PATENT

FOR

CONNECTING A VIRTUAL TOKEN TO A PHYSICAL TOKEN

Inventor: David W. Grawrock

Prepared by: Jeffrey B. Huter
Patent Attorney

intel®
Intel Corporation
5000 W. Chandler Blvd., CH6-404
Chandler, AZ 85226-3699
Phone: (480) 554-4198
Facsimile: (480) 554-7738

"Express Mail" Label Number EL371005724US

CONNECTING A VIRTUAL TOKEN TO A PHYSICAL TOKEN

BACKGROUND

[0001] Existing software-based security services make the implicit assumption that a computing device or platform is trusted. They provide application-level security on the assumption that they execute in a safe environment. This assumption is true enough to justify the level of security required for existing business models, but state-of-the-art security functions are already providing the highest levels of protection that are possible without additional hardware support.

[0002] To this end, the Trusted Platform Computing Alliance (TPCA) describes in the TCPA Main Specification, Version 1.1, 31 July 2001 a Trusted Platform Module (TPM) or physical token that provides increased confidence and that enables enhancements of existing services and new services. The TPM supports auditing and logging of software processes, platform boot integrity, file integrity, and software licensing. The TPM provides a protected information store for the platform that can be used to attest to the identity of the platform as defined by the hardware that is present (e.g. processors, chipsets, firmware, etc.). These features encourage third parties to grant the platform access to information that would otherwise be denied.

[0003] The TPM contains an isolated computing engine whose processes can be trusted because they cannot be altered. These processes and the binding of the subsystem to the platform can combine to reliably measure and report the state of the main computing environment inside the platform. The TPM provides a root of trust for the booting of the platform. While it is the Owner's responsibility to provide a safe

operating system for a platform, once the OS has loaded, it can report the loading of untrusted software to the TPM before that untrusted software is loaded. The TPM can therefore report measured data that indicates the current state of the main software environment in the platform. A local or remote entity can simply query the TPM to reliably obtain these measurements and decide whether the platform's behavior enables it to be trusted for the intended purpose. Confidence in the loading of software is improved, because the TPM can attest to the current state of the operating system.

[0004] The TPM may act as a portal to confidential data, and may allow the release or use of that data only in the presence of a particular combination of access rights and software environment. Of course, the protected store of the TPM may be used for any sensitive data, not just identity information. The TPM may export these services to system-level software security services (such as IPSec) that are themselves called as services by ordinary applications. This arrangement enables greater confidence in the identity of a platform, while simultaneously allowing platform anonymity if so desired. Hence, any application that invokes proof of identity can be used with greater confidence and be allowed greater power.

BRIEF DESCRIPTION OF THE DRAWINGS

[0005] The invention described herein is illustrated by way of example and not by way of limitation in the accompanying figures. For simplicity and clarity of illustration, elements illustrated in the figures are not necessarily drawn to scale. For example, the dimensions of some elements may be exaggerated relative to other elements for clarity. Further, where considered appropriate, reference numerals have been repeated among the figures to indicate corresponding or analogous elements.

[0006] FIG. 1 illustrates an example computing device comprising a physical token and a virtual token.

[0007] FIG. 2 illustrates an example physical token and an example virtual token of FIG. 1.

[0008] FIG. 3 illustrates an example trusted operating environment that may be implemented by the computing device of FIG. 1.

[0009] FIG. 4 illustrates an example method of obtaining a credential for the virtual token of FIG. 1.

DETAILED DESCRIPTION

[0010] In the following detailed description, numerous specific details are described in order to provide a thorough understanding of the invention. However, the present invention may be practiced without these specific details. In other instances, well-known methods, procedures, components and circuits have not been described in detail so as not to obscure the present invention. Further, example sizes/models/values/ranges may be given, although the present invention is not limited to these specific examples.

[0011] References in the specification to "one embodiment", "an embodiment", "an example embodiment", etc., indicate that the embodiment described may include a particular feature, structure, or characteristic, but every embodiment may not necessarily include the particular feature, structure, or characteristic. Moreover, such phrases are not necessarily referring to the same embodiment. Further, when a particular feature, structure, or characteristic is described in connection with an embodiment, it is submitted that it is within the knowledge of one skilled in the art to

effect such feature, structure, or characteristic in connection with other embodiments whether or not explicitly described.

[0012] An example computing device 100 shown in FIG. 1 may comprise one or more processors 110. The processors 110 may support one or more operating modes such as, for example, a real mode, a protected mode, a virtual 8086 mode, and a virtual machine mode (VMX mode). Further, the processors 110 may support one or more privilege levels or rings in each of the supported operating modes. In general, the operating modes and privilege levels of a processor 110 define the instructions available for execution and the effect of executing such instructions. More specifically, a processor 110 may be permitted to execute certain privileged instructions only if the processor 110 is in an appropriate mode and/or privilege level.

[0013] The chipset 120 may comprise one or more integrated circuit packages or chips that couple the processors 110 to memory 130, a network interface 140, a physical token 150, a virtual token 160, and other I/O devices 170 of the computing device 100 such as, for example, a mouse, keyboard, disk drive, video controller, etc. The chipset 120 may comprise a memory controller (not shown) for writing and reading data to and from the memory 130. Further, the chipset 120 and/or processors 110 may define certain regions of the memory 130 as protected memory 132 that may be accessed only by the processors 110 when in a particular operating mode (e.g. protected mode) and privilege level (e.g. 0P).

[0014] The network interface 140 generally provides a communication mechanism for the computing device 100 to communicate with remote agents 190 and certification authorities 195 via a network 180. For example, the network interface 140

may comprise a 10 Mb or 100 Mb Ethernet controller, a cable modem, a digital subscriber line (DSL) modem, plain old telephone service (POTS) modem, etc. to couple the computing device 100 to one or more remote agents 190 and/or certificate authorities 195.

[0015] In general, the physical token 150 of the computing device 100 comprises protected storage for integrity metrics, keys and secrets and may perform various integrity functions in response to requests from the processors 110 and the chipset 120. In particular, the physical token 160 may store integrity metrics in a trusted manner, may quote integrity metrics in a trusted manner, may seal secrets to a particular environment (current or future), and may unseal secrets to the environment to which they were sealed. Further, as will be explained later, the physical token 150 may include an identifier or key that uniquely identifies the computing device 100.

[0016] The virtual token 160 performs in a manner similar to the physical token 150 of the computing device 100. However, the virtual token 160 may comprise more protected storage for integrity metrics, keys and secrets since the virtual token 160 may leverage the storage capacity of the memory 130 and the protect memory 132 to store integrity metrics, keys and secrets. Further, the virtual token 160 may perform integrity operations quicker than the physical token 150 since the virtual token 160 may leverage the processing capabilities of the processors 110 to perform such integrity operations.

[0017] As illustrated in FIG. 2, the physical token 150 may comprise one or more processing units 210 that may perform integrity functions for the computing device 100. The physical token 150 may further generate a fixed private key 220 and a corresponding fixed public key 222 in accordance with an asymmetric cryptographic

algorithm such as, for example, the RSA cryptographic algorithm. In an example embodiment, the physical token 150 generates the fixed private/public key pair 220, 222 such that the fixed private key 220 and corresponding public key 222 are unique and immutable once activated.

[0018] The physical token 150 may also be affixed to or incorporated into the computing device 100 to provide some assurance to remote agents 190 that the physical token 150 is associated with only one computing device 100. For example, the physical token 150 may be incorporated into one of the chips of the chipset 120 and/or surface mounted to the main board of the computing device 100. Due to the uniqueness of the fixed physical token 150 and its incorporation into the computing device 100, a remote agent 190 may identify a computing device 100 with some certainty based upon the fixed public key 222 of the physical token 150.

[0019] Besides the fixed private/public key pair 220, 222, the physical token 150 may further generate one or more supplemental private/public key pairs 230, 232 in accordance with an asymmetric cryptographic algorithm. In an example embodiment, the computing device 100 may generate supplemental private/public key pairs 230, 232 as needed whereas the fixed private/public key pair 220, 222 is immutable. Accordingly, the computing device 100 typically provides the fixed public key 222 to only a small trusted group of entities such as, for example, remote agents 190 and certification authorities 195. Further, the computing device 100 typically utilizes its supplemental private/public key pairs 230, 232 for most other encryption, decryption, and digital signing operations to reduce exposure of the fixed public key 222.

[0020] The physical token 150 may further comprise one or more platform configuration registers (PCR registers) 240, 242, 244 that may be used to record and report integrity metrics in a trusted manner. The processing units 210 may support a PCR quote operation that returns a quote or contents of an identified PCR register 240, 242, 244. The processing units 210 may also support a PCR extend operation that records a received integrity metric in an identified PCR register 240, 242, 244. In particular, the PCR extend operation may (i) concatenate or append the received integrity metric to an integrity metric stored in the identified PCR register 240, 242, 244 to obtain an appended integrity metric, (ii) hash the appended integrity metric to obtain an updated integrity metric that is representative of the received integrity metric and previously integrity metrics recorded by the identified PCR register 240, 242, 244, and (iii) store the updated integrity metric in the PCR register 240, 242, 244.

[0021] As used herein, the verb "hash" and related forms refer to performing an operation upon an operand or message to produce a value or a "hash". Ideally, the hash operation generates a hash from which it is computationally infeasible to find a message with that hash and from which one cannot determine any usable information about a message with that hash. Further, the hash operation ideally generates the hash such that determining two messages which produce the same hash is computationally impossible. While the hash operation ideally has the above properties, in practice one way functions such as, for example, the Message Digest 5 function (MD5) and the Secure Hashing Algorithm 1 (SHA-1) generate hash values from which deducing the message are difficult, computationally intensive, and/or practically infeasible.

[0022] The physical token 150 may be implemented in a number of different manners. However, in an example embodiment, the physical token 150 is implemented to comply with the specification of the Trusted Platform Module (TPM) described in detail in the Trusted Computing Platform Alliance (TCPA) Main Specification, Version 1.1, 31 July 2001.

[0023] Still referring to FIG 2, the virtual token 160 may provide virtual or software constructs that provide functionality similar to the physical token 150. In particular, the virtual token 160 may comprise one or more virtual processing units 250 that may perform integrity functions for the computing device 100. The virtual token 160 may further generate a fixed private key 260 and a corresponding fixed public key 262 in accordance with an asymmetric cryptographic algorithm such that the fixed private key 260 and corresponding public key 262 are unique and immutable once activated.

[0024] Besides the fixed private/public key pair 260, 262, the virtual token 160 may also generate one or more supplemental private/public key pairs 270, 272 in accordance with an asymmetric cryptographic algorithm. The virtual token 160 may further comprise one or more virtual PCR registers 280 that may be used to record and report integrity metrics in a trusted manner.

[0025] An example trusted operating environment 300 is shown in FIG. 3. The computing device 100 may utilize the operating modes and the privilege levels of the processors 110 to establish the trusted operating environment 300. As shown, the trusted operating environment 300 may comprise a trusted virtual machine kernel or monitor 310, one or more standard virtual machines (standard VMs) 320, and one or

more trusted virtual machines (trusted VMs) 330. The monitor 310 of the operating environment 300 executes in the protected mode at the most privileged processor ring (e.g. 0P) to manage security and privilege barriers between the virtual machines 320, 330. Further, the monitor 310 may comprise code that implements the functionality of the virtual token 160. Alternatively, the virtual token 160 may be implemented with a separate VT software module.

[0026] The standard VM 320 may comprise an operating system 322 that executes at the most privileged processor ring of the VMX mode (e.g. 0D), and one or more applications 324 that execute at a lower privileged processor ring of the VMX mode (e.g. 3D). Since the processor ring in which the monitor 310 executes is more privileged than the processor ring in which the operating system 322 executes, the operating system 322 does not have unfettered control of the computing device 100 but instead is subject to the control and restraints of the monitor 310. In particular, the monitor 310 may prevent the operating system 322 and its applications 324 from accessing protected memory 132 and the physical token 150.

[0027] The monitor 310 may perform one or more measurements of the trusted kernel 332 such as a hash of the kernel code to obtain one or more integrity metrics, may cause the physical token 150 to extend an identified PCR register 244 with the integrity metrics of the kernel 332, and may record the integrity metrics in an associated PCR log stored in protected memory 132. The monitor 310 may then determine whether or not it trusts the kernel 332 based upon the integrity metrics of the kernel 332. In response to determining that the kernel 332 is trustworthy, the monitor 310 may

establish the trusted VM 330 in protected memory 132 and launch the trusted kernel 332 in the established trusted VM 330.

[0028] Similarly, the trusted kernel 332 may take one or more measurements of an applet or application 334 such as a hash of the applet code to obtain one or more integrity metrics. The trusted kernel 332 via the monitor 310 may then cause the physical token 150 to extend an identified PCR register 244 with the integrity metrics of the applet 334. The trusted kernel 332 may further record the integrity metrics in an associated PCR log stored in protected memory 132. The trusted kernel 332 may determine whether or not it trusts the applet 334 based upon the integrity metrics of the applet 334. In response to determining that the applet 334 is trustworthy, the trusted kernel 332 may launch the trusted applet 334 in the established trusted VM 330 of the protected memory 132.

[0029] In response to initiating the trusted operating environment 300 of FIG. 3, the computing device 100 also records integrity metrics for the monitor 310 and the virtual token 160 in the monitor PCR register 240 and also records integrity metrics for the hardware in the trusted support (TS) PCR register 242. The trusted operating environment 300 may be initiated in response to various events such as, for example, system startup, an application request, an operating system request.

[0030] In an example embodiment, the computing device 100 obtains and records integrity metrics for the monitor 310 and the virtual token 160 as follows. The processor 110 hashes a monitor software module to obtain a monitor integrity metric. The processor 110 then causes the physical token 150 to extend the monitor PCR register 240 with the monitor integrity metric, and records the monitor integrity metric in

a monitor PCR log stored in protected memory 132. Further, if the virtual token (VT) functionality is implemented as a separate software module, the processor 110 hashes the VT software module to obtain a VT integrity metric. The processor 110 then causes the physical token 150 to further extend the monitor PCR register 240 with the VT integrity metric and records the VT integrity metric in the monitor PCR log. The monitor PCR register 240 now comprises a value that is representative of both the monitor 310 and the virtual token 160.

[0031] The computing device 100 may further record integrity metrics for the hardware in TS PCR register 242 to provide a record of the hardware support for trusted operating environments. In an example embodiment, the processor 110 may obtain hardware identifiers such as, for example, processor family, processor version, processor microcode version, chipset version, and physical token version of the processors 110, chipset 120, and physical token 150. The processor 110 may then extend the TS PCR register 242 with obtained hardware identifiers and record the hardware identifiers in a TS PCR log stored in protected memory 132.

[0032] Besides recording integrity metrics in the physical token 150, the computing device 100 may report the recorded integrity metrics to a remote agent 190 or certification authority 195 to enable the remote agent 190 and certification authority 195 to make trust determinations based upon the reported integrity metrics. For example, the computing device 100 may provide a certification authority 195 with integrity metrics in order to have the certification authority 195 issue the computing device 100 an identity credential that attests to the public key of the identity credential being bound to the entity (e.g. the computing device 100) identified by identity

credential. The computing device 100 may provide remote agents 190 with the issued identity credential in order to obtain secrets from the remote agent 190. The remote agent 190 may then determine whether to release secrets to the entity identified by the identity credential. If the remote agent 190 determines to release the secrets to the entity identified by the identity credential, the remote agent 190 may encrypt the requested secrets with the public key of the identity credential, and provide the requesting entity with the encrypted secrets.

[0033] In an example embodiment, a certification authority 195 may issue a physical token (PT) identity credential to a supplemental public key 232 of the computing device 100. In issuing the PT identity credential, the certification authority 195 basically attests that the supplemental public key 232 of the PT identity credential belongs to the physical token 150 identified by the PT identity credential. The PT identity credential may comprise an identity label for the PT identity credential, the supplemental public key 232 that is bound to this PT identity credential, a reference to a PT endorsement credential, a reference to a platform credential, a reference to a conformance credential, and a version identifier for the physical token 150 that are encrypted by a private key CAPriv of the issuing certification authority 195.

[0034] The PT endorsement credential, the platform credential, and the conformance credential referenced by the PT identity credential are credentials that certificate authorities 195 issue to attest to integrity aspects of the computing device 100 and to provide information in support of the attestation. In particular, the PT endorsement credential attests that the fixed public key 232 of the PT endorsement credential was generated by the identified physical token 150 and that the identified

physical token 150 satisfies criteria defined for suitable physical tokens 150. Similarly, the platform credential basically attests that the referenced PT endorsement credential is bound to a platform or computing device 100 having a design and construction that satisfies criteria defined for suitable platforms. Further, the conformance credential attests that the referenced PT endorsement credential is bound to a platform or computing device 100 having an overall design that conforms to criteria defined for suitable platforms.

[0035] The PT identity credential, the PT endorsement credential, the platform credential, and the conformance credential may be implemented and may be issued in a number of different manners. However, in an example embodiment, the PT identity credential, the PT endorsement credential, the platform credential, and the conformance credential are implemented and issued in a manner that respectively complies with the Identity Credential, the TPM Endorsement Credential, the Platform Credential, and the Conformance Credential described in detail in the TCPA Main Specification, Version 1.1, 31 July 2001.

[0036] Referring now to FIG. 4, there is depicted a method 400 of obtaining a VT endorsement credential from a selected certification authority 195. The computing device 100 may execute the method 400 in response to various events such as, for example, the monitor 310, the standard VM 320, the trusted VM 330, a remote agent 190, and/or a certification authority 195 requesting that the computing device 100 establish a virtual token 160 and a corresponding VT endorsement credential. In issuing the VT endorsement credential, the selected certification authority 195 attests that fixed public key 262 of the VT endorsement credential is bound to the virtual token

160, the virtual token 160 is bound to and uniquely identifies the computing device 100, and the virtual token 160 satisfies criteria defined for suitable virtual tokens.

[0037] The computing device 100 and the selected certification authority 195 may perform all or a subset of the method 400 in response to executing instructions of a machine readable medium such as, for example, read only memory (ROM); random access memory (RAM); magnetic disk storage media; optical storage media; flash memory devices; and/or electrical, optical, acoustical or other form of propagated signals such as, for example, carrier waves, infrared signals, digital signals, analog signals. Furthermore, while the method 400 illustrates operation of the computing device 100 and the selected certification authority 195 as a sequence of operations, the computing device 100 and the selected certification authority 195 may perform various operations in parallel or in a different order.

[0038] In block 410, the monitor 310 generates and/or obtains parameters that are provided to the physical token 150 in response to requesting the physical token 150 to attest to the identity of the monitor 310, to tie the received parameters to the request, and to attest to the current hardware and/or software environment of the computing device 100. The monitor 310 in block 418 may make such a request of the physical token 150 by executing a request function such as, for example, a TPM_VirtualTPMCredential function. The format of the example TPM_VirtualTPMCredential function is shown below:

TPM_VirtualTPMCredential (SigH, MQuote, CertM1, H2, AllPCRQuote)

where the signed hash SigH is an input parameter and the parameters MQuote, CertM1, H2, and AllPCRQuote are output parameters provided by the physical token 150.

[0039] The signed hash SigH provides the physical token 150 with a value that identifies the VT public key 262, a PT identity credential PTIdCred of the physical token 150, and a public key CAPub of the selected certification authority 195. For example, the monitor 310 may generate a data set that includes the VT public key 262, the PT identity credential PTIdCred, and the public key CAPub and may encrypt the data set with the VT private key 260 to obtain the signed data set SigDS. The monitor 310 may then hash the signed data set SigDS via the SHA-1 hash algorithm or some other algorithm to obtain the signed hash SigH that attests to, fingerprints, or identifies the VT public key 262, the PT identity credential PTIdCred, and the public key CAPub.

[0040] In response to the request, the physical token 150 in block 422 provides the monitor 310 with information that attests to the identity of the monitor 310, the VT public key 262, and PT identity credential PTIdCred and that attests to the current hardware and/or software environment of the computing device 100. For example, in response to the example TPM_VirtualTPMCredential provides the monitor 310 with a certification message CertM1, a signed index MQuote, quotes AllPCRQuote of all PT PCR registers 240, 242, 244, and a hash H2 that provides uniqueness and attests to other parameters. The physical token 150 may generate the signed index MQuote by encrypting the index MPCR that identifies the monitor PCR register 240 of the physical token 150 with the supplemental private key 230 associated with the PT identity credential PTIdCred.

[0041] In an example embodiment, the physical token 150 generates the certification message CertM1 based upon the following message format:

CertM1 = (Format1, H1, H2) CAPub

which indicates the certification message CertM1 comprises a hash H1 and the hash H2 encrypted by the public key CAPub. The format parameter Format1 may comprise a predefined ASCII string such as "PT_CertM1" that identifies the message as a certification message CertM1 created by the physical token 150. The public key CAPub corresponds to the public key of the selected certification authority 195 which may have been provide by the request of the monitor 310 or previously provided to the physical token 150.

[0042] The hash H1 identifies the VT public key 262, the PT identity credential PTIdCred, and the public key CAPub and provides uniqueness to help prevent replay attacks. In an example embodiment, the physical token 150 generates a nonce Nonce1 such as a 160 bit random value generated by a random number generator of its processing unit 210. The physical token 150 may then generate the hash H1 by hashing the signed hash SigH received from the monitor 310 and the generated nonce Nonce1 according to a hashing algorithm such as the SHA-1 hashing algorithm.

[0043] The hash H2 attests to the hash H1, the index MPCR, the signed index MPCR, and provides uniqueness. In an example embodiment, the physical token 150 may generate the hash H2 by hashing the hash H1, the index MPCR, and the signed index MQuote, thus tying the hash H1 to the index MPCR and the signed index MQuote. Further, since the hash H2 includes the hash H1, the hash H2 further attests to the VT public key 262, the PT identity credential PTIdCred, and the public key

CAPub, thus further tying these parameters to the index MPCR and the signed index MPCR. Further, the hash H2 comprises uniqueness through the nonce Nonce1 of the hash H1.

[0044] The monitor 310 in block 426 may generate a credential request package CredP1 that provides the certification authority 195 with information from which it may decide to issue a VT endorsement credential. In an example embodiment, the physical token 150 generates the credential request package CredP1 based upon the following package format:

$$CredP1 = (EDS1, CredM1 \bullet CAPub)$$

which indicates the credential request package CredP1 comprises a credential request message CredM1 encrypted with the public key CAPub of the selected certification authority 195 and an encrypted data set EDS1 that is not encrypted with the public key CAPub.

[0045] The encrypted data set EDS1 provides information about the computing device 100. In an example embodiment, the monitor 310 generates a data set DS1 that includes the certification message CertM1, the VT public key 262, the PT identity credential PTIdCred, an index TSPCR that identifies the TS PCR register 242, the quotes AllPCRQuote of the PCR registers 240, 242, 244, and logs AllPCRLogs of the PCR registers 240, 242, 244. The monitor 310 further generates a session key S1 and encrypts the data set DS1 using the session key S1 and a symmetric encryption algorithm such as, for example, DES, 3DES, and/or AES.

[0046] The credential request message CredM1 attests to the information of the encrypted data set EDS1 and provides the session key S1 used to encrypt/decrypt the

encrypted data set EDS1. In an example embodiment, the monitor 310 generates the credential request message CredM1 based upon the following message format:

CredM1 = (Format2, H2, H3, S1) CAPub

which indicates the credential request message CredM1 comprises a format parameter Format2, the hash H2 provided by the physical token 150, another hash H3, and the session key S1 encrypted with the public key CAPub of the selected certification authority 195.

[0047] The format parameter Format2 may comprise a predefined ASCII string such as "Mon_CredM1" that identifies the message as a credential request message created by the monitor 310. The hash H3 attests to the data of the data set DS1. The monitor 310 may generate the hash H3 by hashing the parameters of the data set DS1 using a hashing algorithm or some other algorithm. In an example embodiment, the monitor 310 hashes the certification message CertM1, the VT public key 262, the PT identity credential PTIdCred, the index TSPCR, the quotes AllPCRQuote, and the logs AllPCRLogs using the SHA-1 hashing algorithm.

[0048] In block 430, the monitor 310 may request the selected certification authority 195 to issue a VT endorsement credential to the virtual token 160 and may transfer the credential request package CredP1 to the selected certification authority 195 in support of the request. In an example embodiment, the monitor 310 transfers the credential request package CredP1 to the certification authority 195 as part of a transaction that may provide further information about the request such as encryption algorithms used, hashing algorithm used, credential request package format used, etc.

[0049] The certification authority 195 in block 434 may unpack the credential request package CredP1 to obtain information provided by the credential request package CredP1. For example, the certification authority 195 may decrypt the credential request package CredP1 using its corresponding private key CAPriv to obtain the session key S1 and the other information of the credential request package CredP1. The certification authority 195 may further decrypt the encrypted data set EDS1 using the session key S1 of the certification message CertM1 to obtain the data of the data set DS1. The certification authority 195 may further decrypt the certification message CertM1 and the credential request message CredM1 using its private key CAPriv.

[0050] In block 438, the certification authority 195 may validate the information provided by the credential request package CredP1. For example, the certification authority 195 may re-calculate hashes H1, H2, H3 from parameters provided by the data set DS1 to obtain computed hashes CH1, CH2, CH3. The certification authority 195 may then determine that the data of the data set DS1 are the same parameters that the hashes H1, H2, H3 attested in response to the computed hashes CH1, CH2, CH3 having a predetermined relationship (e.g. equal) with the provided hashes H1, H2, H3.

[0051] In response to determining that the information provided by the credential request package CredP1 do not correspond to the hashes H1, H2, H3 of the credential request package CredP1, the certification authority 195 in block 442 may refuse to issue the requested VT endorsement credential. The certification authority 195 may further provide the computing device 100 with an indication as to why the request for the VT endorsement credential VTCred was refused. In an example embodiment, the

certification authority 195 does not provide any indication as to why the request was refused to prevent supplying information that may be used to circumvent the certification process.

[0052] In response to determining that the information provided by the credential request package CredP1 corresponds to the hashes H1, H2, H3 of the credential request package CredP1, the certification authority 195 in block 446 may determine whether it distrusts the physical token 150 based upon the PT identity credential parameter PTIdCred. The certification authority 195 may determine that the virtual token 160 is not trustworthy because the identified version of the physical token 150 has known vulnerabilities, the PT identity credential has expired, and/or the entity identified by the PT identity credential is untrustworthy. The certification authority 195 may further determine that the virtual token 160 is not trustworthy because the certification authority 195 that issued the PT identity credential is untrustworthy, the referenced platform conformance credential is untrustworthy, and/or the referenced conformance credential is untrustworthy.

[0053] In response to distrusting the virtual token 160 based upon the PT identity credential parameter PTIdCred, the certification authority 195 in block 442 may refuse to issue the requested VT endorsement credential. Otherwise, the certification authority 195 in block 450 may determine based upon the quote of the monitor PCR register 240 and its corresponding PCR log whether the certification authority 195 distrusts the virtual token 160 of the computing device 100. As indicated above, the monitor PCR register 240 records integrity metrics of the monitor 310 and the virtual token 160. Further, the monitor PCR log comprises the integrity metrics recorded by

the monitor PCR register 240. Accordingly, the certification authority 195 may obtain the quote and log of the monitor PCR register 240 from the quote AllPCRQuote and logs AllPCRLog based upon the index MPCR of the credential request package CredP1. The certification authority 195 may then determine whether the quote and log of the monitor PCR register 240 are indicative of a trusted implementation of the monitor 310.

[0054] In response to distrusting the virtual token 160 based upon the quote and/or log of the monitor PCR register 240, the certification authority 195 in block 442 may refuse to issue the requested VT endorsement credential. Otherwise, the certification authority 195 in block 454 may determine based upon the quote of the TS PCR register 242 and its corresponding PCR log whether the certification authority 195 distrusts the virtual token 160 of the computing device 100. As indicated above, the TS PCR register 242 records integrity metrics for the hardware components of the computing device 100 from which trusted environment support capabilities may be determined. Further, the TS PCR log comprises the integrity metrics recorded by the TS PCR register 242. Accordingly, the certification authority 195 may obtain the quote and log of the TS PCR register 242 from the quotes AllPCRQuote and the logs AllPCRLog based upon the index TSPCR of the credential request package CredP1. The certification authority 195 may then determine whether the quote and log of the TS PCR register 242 are indicative of trusted hardware components.

[0055] In response to distrusting the virtual token 160 based upon the quote and/or log of the TS PCR register 242, the certification authority 195 in block 442 may refuse to issue the requested VT endorsement credential. Otherwise, the certification

authority 195 in block 458 may determine based upon additional information of the credential request package CredP1 whether to distrust the virtual token 160 of the computing device 100. For example, the certification authority 195 may determine to distrust the virtual token 160 based upon the quotes and/or logs of the other PCR registers 244 provided by the quotes AllPCRQuote and the logs AllPCRLog of the credential request package CredP1.

[0056] In response to distrusting the virtual token 160 based upon the additional information, the certification authority 195 in block 442 may refuse to issue a VT endorsement credential. Otherwise, the certification authority 195 in block 462 may provide the monitor 310 with the requested VT endorsement credential. To this end, the certification authority 195 may generate the VT endorsement credential to include a label identifying the credential as a VT endorsement credential, the fixed public key 262 of the virtual token 160, a reference to the certification authority 195, and possibly other information about the computing device 100, the physical token 150, and/or virtual token 160. The certification authority 195 may then digitally sign the VT endorsement credential with its private key CAPub and further encrypt the signed endorsement credential with the public key 262 of the virtual token 160.

[0057] The computing device 100 may later provide a remote agent 190 with the obtained VT endorsement credential to attest that the fixed public key 262 of the VT endorsement credential is bound to the computing device 100. Furthermore, the computing device 100 may utilize the VT endorsement credential to obtain VT identity credentials from a selected certification authority 195 that attest to a supplemental public key 272 being bound to the computing device 100. The certification authority

195 may issue the VT identity credentials in a manner similar to the PT identity credentials described above.

[0058] It should be appreciated that the above method 400 is merely an example method. For example, the method 400 utilizes techniques to overcome size limits of typical asymmetric cryptographic algorithms. Asymmetric cryptographic algorithms generally only successfully encrypt data or data sets having a size that is less than or equal to a maximum size defined by a modulus of the asymmetric cryptographic algorithm. One way to overcome the size limit of an asymmetric cryptographic algorithm is to generate a session key for a symmetric algorithm, encrypt the data set using the session key and the symmetric algorithm, and encrypt the session key using the an asymmetric key and an asymmetric encryption algorithm. The data set may be later decrypted by the session key which is retrieved using the corresponding asymmetric key and asymmetric algorithm. Thus, in effect encrypting the data set via the asymmetric key and asymmetrically encryption algorithm.

[0059] Another way to overcome the size limit of an asymmetric cryptographic algorithm is to generate one or more hashes that attest to the data contained in a data set and asymmetrically encrypt the one or more generated hashes. An entity (e.g. the certification authority 195) may asymmetrically decrypt the hashes, calculate computed hashes from a received data set, and determine that the received data set is the same data set attested to by the decrypted hashes in response to the computed hashes having a predetermined relationship (e.g. equal) with the decrypted hashes.

[0060] Yet another way to overcome the size limit is to split the data set into data subsets that are less than the maximum size of an asymmetric cryptographic algorithm.

The data subsets may then be individually encrypted using an asymmetric key and the asymmetric algorithm. An entity (e.g. the certification authority 195) may later asymmetrically decrypt each of the subsets using the corresponding asymmetric key and reconstruct the data set from the data subsets.

[0061] It should be appreciated that depending upon the size of the data sets and size limits of the encryption algorithms used, the method 400 may be implemented to use zero or more of the above techniques. Furthermore, it should be appreciated that the method 400 as described also uses hashes (e.g. H1, H2, H3) that are compact representations (e.g. 160 bit) of larger data sets (160 kilobytes) to reduce traffic between components (e.g. monitor and physical token 150) and/or entities (computing device 100 and certification authority 195). However, it should be appreciated that, in general, generation and/or transfer of the hashes may be replaced with transfers of the data sets represented by the hashes.

[0062] For example, the monitor 310 may provide the physical token 150 with one or more portions of the data set (e.g. the VT public key 262, the PT identity credential PTIdCred, and/or the public key CAPub) represented by the signed hash SigH. Thus, the generation and transfer of the signed hash SigH or the inclusion of those portions of the data set in the signed hash SigH may be eliminated. The physical token 150 may then attest to the received portions of the data set via the hashes H1, H2 of the certification message CertM1 as described above. Alternatively, the physical token 150 may attest to the received portions of the data set by including one or more of them in the certification message CertM1. Thus, the hashes H1, H2 of the certification message CertM1 may be eliminated or may be generated without including

portions of the data set that are attested to by mere inclusion in the certification message CertM1. It should be appreciated that if the certification message CertM1 comprises data instead of a hash of the data, such data may be eliminated from the data set DS1 since certification authority 195 may obtain the data from the certification message CertM1.

[0063] While certain features of the invention have been described with reference to example embodiments, the description is not intended to be construed in a limiting sense. Various modifications of the example embodiments, as well as other embodiments of the invention, which are apparent to persons skilled in the art to which the invention pertains are deemed to lie within the spirit and scope of the invention.